


Fast Rerouting Against Dynamic Failures: 2-Resilience via Ear-Decomposition and Planarity

Wenkai Dai ✉ 

TU Berlin, Germany

Faculty of Computer Science and UniVie Doctoral School Computer Science DoCS, University of Vienna, Austria

Klaus-Tycho Foerster ✉ 

Department of Computer Science, Technical University of Dortmund, Germany

Stefan Schmid ✉ 

TU Berlin and Fraunhofer SIT, Germany

Abstract

Modern communication networks employ local fast failover mechanisms in the data plane, swiftly reacting to link failures through pre-installed rerouting rules. This paper investigates resilient routing schemes that guarantee packet delivery under up to k link failures, provided the source and destination remain connected in the degraded network. While prior theoretical studies have mainly addressed static failures, where multiple links fail simultaneously and permanently, real networks often experience dynamic failures, such as transient link flapping caused by short-lived faults.

We study the limits of basic and source-matched failover routing with packet-header rewriting against dynamic failures in general graphs. In basic routing, forwarding depends only on active links, incoming ports, and the destination, whereas source-matched routing additionally incorporates the source, requiring more memory (and logic) at the router. The 2-resilient source-matched routing for static failures is shown to fail under permanent but non-simultaneous failures. Moreover, even with source matching, we prove that in planar graphs $k \geq 2$ resilience is impossible without bit rewriting, and in general graphs, perfect k -resilience is unachievable by only rewriting $O(\log k)$ bits.

For planar graphs, we introduce ear-decomposition into basic routing and develop novel local rerouting mechanisms that tolerate dynamic failures. These yield tight 2-resilient basic routing by rewriting only one or two bits, closing the gap between lower bounds and practical routing scheme.

2012 ACM Subject Classification Networks → Routing protocols; Computer systems organization → Dependable and fault-tolerant systems and networks

Keywords and phrases Resilience, Local Failover, Routing, Dynamic Link Failures, Link Flapping

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2025.20

Related Version *Preliminary Version: On the Resilience of Fast Failover Routing Against Dynamic Link Failures* [9]

Funding German Research Foundation (DFG) project ReNO, Schwerpunktprogramm: Resilienz in Vernetzten Welten – Beherrschen von Fehlern, Überlast, Angriffen und dem Unbekannten (SPP 2378), 2023–2027.

1 Introduction and Related Work

Communication networks are a critical infrastructure of the digital society. Since link failures—inevitable and increasingly frequent in large-scale systems [18]—can severely degrade service [1, 30, 35], modern networks deploy local fast failover in the data plane to respond faster to failures. These mechanisms rapidly reroute packets along preinstalled backup paths, avoiding global recomputation and enabling recovery times orders of magnitude faster [13, 22].

The core challenge of devising efficient data-plane failover mechanisms is to precompute local rerouting rules ensuring reachability under link failures, without knowledge of



© W. Dai, K.-T. Foerster, and S. Schmid;

licensed under Creative Commons License CC-BY 4.0

29th International Conference on Principles of Distributed Systems (OPODIS 2025).

Editors: Andrei Arusaie, Emanuel Onica, Michael Spear, and Sara Tucci-Piergiovanni; Article No. 20;

pp. 20:1–20:20



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

■ **Table 1** Summary of related work and our contributions: prior results appear in white–gray (static failures) and green (dynamic failures) rows, while our new results are highlighted in blue.

Failure Type	Rewriting Bits #	Routing information		Graph Classes	Resilience results for static & deterministic routing
		<i>Per-source</i>	<i>Per-dest.</i>		
static	no		✓	●	$(k-1)$ -resilience for $k \leq 5$, open for $k \geq 6$ [5]
static	no		✓	●	$(\lfloor k/2 \rfloor)$ -resilience for any k [5]
static	no	✓	✓	●	$(k-1)$ -resilience for any k [14]
static	3		✓	●	$(k-1)$ -resilience [5]
static	no		✓	\triangle	1-resilience [13], no ≥ 2 -resilience (\square) [5]
static	no	✓	✓	\triangle	arbitrary k -resilience impossible [17]
static	no	✓	✓	\triangle	2-resilience possible, 3-resilience impossible [8]
dynamic	no		✓	●	$(k-1)$ -resilience possible for $k \leq 5$ [10]
dynamic	no		✓	●	$(\lfloor k/2 \rfloor)$ -resilience possible for any k [10]
dynamic	$\log k$		✓	●	$(k-1)$ -resilience possible for any k [10]
semi-dynamic	3		✓	●	$(k-1)$ -resilience possible for any k [10]
dynamic	3		✓	●	HDR-3-Brrs in [5, Algorithm 2] inapplicable [10]
dynamic	no		✓	●	$(k-1)$ -resilience imposs. for $k \geq 2$ (link-circular) [10]
dynamic	no		✓	\triangle	1-resilience possible [10], no ≥ 2 -resilience (\square) [5, 10]
dynamic	no	✓	✓	\square	≥ 2 -resilience is impossible [Thm. 3]
dynamic	$\log k$	✓	✓	\triangle	arbitrary k -resilience impossible [Thm. 4]
semi-dynamic	no	✓	✓	\triangle	2-resilient algo. (static failures) in [8] inapplicable [Thm. 2]
dynamic	1 or 2		✓	\square	2-resilient basic routing on planar graphs [Thms. 18 & 16]

Legend: ✓ indicates that this parameter is included in forwarding function. Graph classes are denoted as ● k -edge-connected, \square planar, and \triangle general.

downstream link states. This reduces to the following fundamental question:

- Can we devise a failover routing scheme capable of tolerating any k link failures as long as the underlying topology remains connected?

Resilient failover mechanisms have attracted significant attention [5, 7, 8, 13, 16, 31]. Randomized approaches [3, 6], resembling random walks for graph exploration, can offer resilience but are impractical due to packet reordering and limited router support [14]. Similarly, packet duplication techniques such as flooding impose prohibitive overhead.

Standard routers implement *basic* failover routing by deterministically forwarding packets using only local information such as active links, in-ports, and destinations. Feigenbaum et al. [13] introduced DAG-based failover resilient to a single failure, but *perfect resilience*—tolerance to arbitrarily many failures—is impossible even in small graphs [15]. Chiesa et al. [5] further proved that basic routing cannot achieve 2-resilience even in planar graphs. Consequently, research has explored heuristics, extra local inputs [8, 10], header rewriting [4, 5], or dense connectivity [3, 5, 7, 17, 34]. In particular, header rewriting allows routers to modify reserved header bits to guide subsequent forwarding decisions.

For k -edge-connected graphs, Chiesa et al. [5] showed that $(k-1)$ -resilient basic routing can be efficiently computed for $k \leq 5$, while the case $k > 5$ remains open; with header-rewriting, they devised schemes that achieve $(k-1)$ -resilience in arbitrary k -edge-connected graphs using either $\log k$ or 3 header bits.

However, real networks are often sparse with locally dense subregions [11, 16], necessitating the study of general topologies. Dai et al. [8] showed that 2-resilient routing is feasible in general graphs when forwarding depends on both source and destination (*source-matched* routing), but ≥ 3 -resilience is impossible. Source-matched routing, however, incurs up to n -fold overhead in computing and storing per-pair routing tables compared to basic routing.

Most theoretical work assumes *simultaneous fail-stop failures*, where links fail once and remain unavailable, i.e., static graphs. In practice, however, failures are often *asynchronous*,

with links recovering or *flapping* [24, 27, 29, 32]. Gill et al. [18] further classify datacenter link failures as *long-lived* or *sporadic short-lived*, the latter typically caused by software or transient errors.

As such dynamics turn the network into a time-varying graph, invalidating conventional failover routing schemes, Dai et al. [10] model link failures into three types: *dynamic*, where unstable links arbitrarily switch between up and down states; *semi-dynamic*, where fail-stop links may fail during packet traversal; and *static*, where all fail-stop links fail simultaneously. They revisit the ideal resilience results of [5] under dynamic settings and show that certain established results, e.g., the 3-bit rewriting, may become ineffective (see Tab. 1 for details).

Hence, this paper investigates deterministic worst-case k -resilience on general graphs under the dynamic link failure models of [10], focusing on basic routing augmented with packet-header rewriting. Our work is closely related to Dai et al. [8, 10] and Chiesa et al. [5], as we extend the results of [8] to dynamic failures only on basic routing functions.

We show that in dynamic settings, the power of source-matched routing collapses to that of basic routing.

For instance, although 2-resilient source-matched routing is always achievable under static failures in general graphs [8], it becomes impossible even for planar graphs (see the counterexample in Fig. 4), as backtracking to the original source may be infeasible.

At the same time, we resolve the open question of [5] by providing the first 2-resilient basic routing on planar graphs by rewriting one- or two-bit in the packet header, even under dynamic failures. An overview of related results and our contributions is given in Table 1.

1.1 Contributions

This paper initiates the study of perfect resilience of fast rerouting under dynamic, non-simultaneous link failures. A summary of our results is given in Table 1. We prove that the 2-resilient source-matched routing of Dai et al. [8] already fails under semi-dynamic failures, and that 2-resilience via source-matched routing is impossible in planar graphs without packet-header rewriting. This impossibility extends to k -resilience in general graphs, even with $O(\log k)$ rewritable bits. In contrast, we present the first algorithms achieving 2-resilience against dynamic failures in planar graphs using basic routing by rewriting only one or two header bits. This is tight with respect to our dynamic lower bounds stated in Theorem 3 and the static lower bound of [5], thereby resolving the open problem of [5]. We can compute the routing scheme in $O(n + m)$ time per destination on planar graphs, whereas source-matched routing [8] needs $O(nm)$ per source-destination pair, i.e., $\Omega(n^2)$ times slower.

Technical Novelty We introduce *ear-decomposition* into basic routing to design ear-based schemes that mimic source-matched routing, where each ear serves as a local source without interfering with others. Unlike existing approaches [5, 8, 17], which detour along induced cycles containing the failure (and thus risk looping under dynamic failures), our method selects *landmark nodes* to terminate such detours independently of the failure. Our algorithms leverage structural properties of *planar embeddings*, and we believe these techniques may be of independent interest for future studies on resilience against dynamic failures.

1.2 Organization

The remainder of this paper is organized as follows. We introduce our formal model in §2, then discuss the limitations of prior methods [5, 8] and present impossibility results for k -resilient routing by rewriting bits under dynamic failures in §3. Next, we develop algorithms

for computing 2-resilient basic routing schemes on planar graphs via one- or two-bit rewriting in §4, and conclude in §5 with open questions. We defer lengthy proofs to Appendix A–B.

2 Models and Preliminaries

We model the network as an *undirected simple* $G = (V, E)$, where vertices in V represent routers and undirected edges $\{u, v\} \in E$ represent *bi-directed* links. For $E' \subseteq E$, let $G \setminus E' = (V, E \setminus E')$; for $V' \subseteq V$, $G \setminus V'$ denotes the graph obtained by removing V' and all *incident edges*. For a subgraph $G' \subseteq G$, we write $N_{G'}(v)$, $E_{G'}(v)$, and $\Delta_{G'}(v)$ for the *neighbors*, *incident edges*, and *degree* of v in G' , respectively, omitting the subscript G' when clear from context. Each undirected edge $\{u, v\} \in E$ corresponds to two *directed arcs* (u, v) and (v, u) . For $E' \subseteq E$ and $v \in V$, we slightly abuse notation to let E'_v denote the subset of edges in E' that are incident to v . For multigraphs, we extend the graph definition appropriately by allowing multiple distinguishable edges between any two vertices in V .

Failure Models. Let $F \subseteq E$ denote a set of *link failures* in G , where each $e \in F$ may fail to transfer packets in both directions when its state is *down* (failed). A link is *fail-stop* if its down state is permanent. We classify F as:

- *Static* — all links in F are fail-stop and fail simultaneously;
- *Semi-dynamic* — all links in F are fail-stop but may fail at different times;
- *Dynamic* — links in F may alternate arbitrarily between *up* and *down* states over time.

Failover Routing Functions. In failover routing, each node $v \in V$ stores a *predefined, static forwarding (interchangeably, routing) function* to *deterministically* select an *outgoing link* (out-port) for each incoming packet based solely on local information at v . Formally, given a graph G , a *general routing function* at $v \in V$ is

$$\pi_{G,v}: V \times (V \cup \{\perp\}) \times (N_G(v) \cup \{\perp\}) \times 2^{E_G(v)} \rightarrow E_G(v),$$

where V is the destination $t \in V$ of the incoming packet; $V \cup \{\perp\}$ gives the source of the incoming packet, with \perp meaning “no source information,”; $N_G(v) \cup \{\perp\}$ is the incoming link (in-port) at v , with \perp indicating packet origination at v ; and $2^{E_G(v)}$ is the set of currently active (non-failed) links incident to v . For multigraphs, $\pi_{G,v}$ extends naturally.

The routing function $\pi_{G,v}$ is *basic* if the source is always \perp , and *source-matched* otherwise. Unless stated, we consider basic routing. Fixing a destination $t \in V$ (basic) or a pair of source-destination $(s, t) \in V \times V$ (source-matched), we write $\pi_{G,v}^t$ and $\pi_{G,v}^{(s,t)}$, omitting G when clear. A *routing scheme* is the collection of routing functions on V , e.g., $\Pi^t = \bigcup_{v \in V \setminus \{t\}} \pi_v^t$.

It is worth noting that Feigenbaum et al. [13] showed that 1-resilience against static failures is impossible when using fewer parameters than those available in basic routing.

Header-Rewriting Routing. Header-rewriting augments a routing scheme by reserving $k \in \mathbb{N}$ rewritable bits in each packet’s header. A routing function π at $v \in V$ can interpret and modify these bits to influence the current and subsequent forwarding decisions:

$$\text{HDR-}k\text{-}\pi: \text{dom}(\pi) \times \{0, 1\}^k \rightarrow E_G(v) \times \{0, 1\}^k.$$

In practice, k is limited by competing fields (e.g., TTL, checksums, QoS). Bit-rewriting adds processing overhead, latency, and potential packet loss, so minimizing k improves transmission efficiency.

Further Concepts in Failover Routing. We introduce several commonly used notions in failover routing. A packet is said to *get stuck* if it halts at a node other than its destination. For deterministic routing without header bit rewriting, a *forwarding loop* occurs when a packet traverses the same *directed* link (arc) twice; both directions of an undirected link may each be traversed once without forming a loop. Any loop possible under static failures can also occur under semi-dynamic or dynamic failures. A packet fails to reach its destination t if it either gets stuck or enters a forwarding loop. A routing function at node v is *link-circular* if v forwards packets through an *ordered circular sequence* $\langle u_1, \dots, u_\ell \rangle$ of neighbors, forwarding from u_i to u_{i+1} (indices taken modulo ℓ). If link $\{v, u_{i+1}\}$ fails, the packet is sent to u_{i+2} , and so forth, cycling back to u_1 after u_ℓ [5].

► **Definition 1** (*k-Resilient Failover Routing Problem*). *Let $G = (V, E)$ be a graph and let $t \in V$ be a designated destination. The k -resilient failover routing problem is to compute a k -resilient routing scheme for t in G under the static, semi-dynamic, or dynamic failure models. A forwarding scheme for t is k -resilient if, for every source $s \in V$ and every set of link failures $F \subseteq E$ with $|F| \leq k$, the scheme delivers a packet from s to t in the given failure model whenever s and t remain connected in $G \setminus F$. Here, $G \setminus F$ denotes the subgraph obtained by removing all edges in F , independent of the failure model.*

Note on Definition 1. The subgraph $G \setminus F$ may have multiple connected components. We assume that dynamic or semi-dynamic failures in F *do not* cause the routing functions to forward packets between distinct components of $G \setminus F$; such cases are beyond the scope of our resilient routing model. Static failures are a subset of semi-dynamic failures, which in turn are a subset of dynamic failures; a resilient routing algorithm for one failure type applies to its subsets, while impossibility results extend to its supersets. Since resilient routing may require packets to retrace paths in reverse, dynamic and semi-dynamic failures can introduce *inconsistencies* in the set of active links perceived by deterministic routing functions, thereby increasing the likelihood of forwarding loops.

We focus on computing a k -resilient routing scheme for a fixed destination t , as the same algorithm applies to any $u \in V$. For source-matched routing, it suffices to consider a fixed pair (s, t) , since the algorithm generalizes to any $s, t \in V$.

Further Notations and Graph Theory Concepts. We first introduce the graph-theoretic concepts and notations used in this chapter. A path P from u to v in G is called a u - v path. Two paths are *edge-disjoint* if they share no edges (but may share vertices), and two edge-disjoint u - v paths P_1 and P_2 are *node-disjoint* if $V(P_1) \cap V(P_2) \setminus \{u, v\} = \emptyset$.

We focus on *edge-connectivity*, referred to simply as *connectivity*. For $u, v \in V$, two nodes u and v are (interchangeably, $u - v$ is) *k-edge-connected* (or *k-connected*) if there exist k edge-disjoint u - v paths in G . The graph G is *k-edge-connected* (or *k-connected*) if any two distinct nodes are k -connected. Nodes u and v are *exactly k-connected* if they are k -connected but not $(k + 1)$ -connected.

Given a path $P = (x_0, x_1, \dots, x_k)$, with $x_i \in V(P)$ for $0 \leq i \leq k$, the notation $x_i P x_j \subseteq P$ denotes the subpath of P from x_i to x_j for $0 \leq i < j \leq k$. For $V' \subseteq V$, the *induced subgraph* $G[V']$ contains all edges $\{u, v\} \in E$ with $u, v \in V'$.

A graph $G = (V, E)$ is *planar* if it can be drawn on the Euclidean plane \mathbb{R}^2 without two edges intersecting except possibly at a common endpoint; such a drawing is a *plane embedding* of G . Given a plane embedding, a *face* is a connected component of $\mathbb{R}^2 \setminus G$ whose boundary consists of vertices and edges of G . The unique unbounded face is the *outer face* (or *unbounded face*), and all other faces are *inner (interior) faces* (or *bounded faces*).

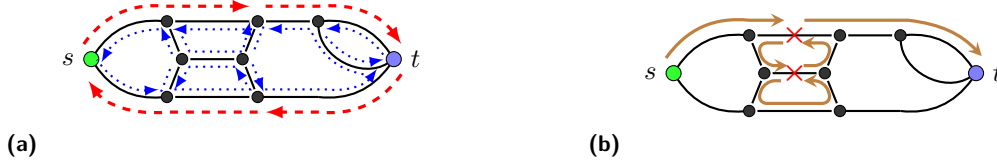


Figure 1 The main idea of 2-resilient source-matched routing scheme [8] computes a special planar subgraph $\mathcal{G} \subseteq G$ in which both s and t lie on the *outer face*, and every inner face shares at least one edge with the outer face. The routing rules are simple: packets traverse the outer face counterclockwise (red dashed arrows in Fig. 1a) and, upon encountering a failure, detour clockwise around the corresponding inner face (blue dotted arrows). As illustrated in Fig. 1b, a packet starting at s first follows the directed s - t path along the outer face. If an arc (u, v) fails, the packet is rerouted along the directed path that includes the reverse (failed) arc (v, u) . This approach successfully tolerates two *static* failures. However, under *dynamic* failures, the detour may revisit a previously failed edge after it has recovered, leading to routing loops.

3 Limitations of Previous Approaches and Lower Bounds

In this section, we first highlight the limitations of previous failover approaches under dynamic failures, then show that 2-resilient source-matched routing is infeasible even for planar graphs, and finally prove that perfect k -resilience is impossible with only rewriting $O(\log k)$ bits.

3.1 Limitations of Prior Work

Without utilizing rewritable bits in packet headers, Chiesa et al. [5] proved that 2-edge-connected (planar) graphs do not admit 2-resilient routing against static failures. In contrast, Dai et al. [8] developed a 2-resilient routing algorithm for static failures in general graphs by additionally incorporating source matching. However, achieving 3-resilience under static failures is already impossible [8].

The 2-resilient routing algorithm of Dai et al. [8] computes, for each source-destination pair (s, t) in a general graph G , a planar *kernel graph* $\mathcal{G} \subseteq G$ that preserves s - t connectivity under any two edge failures. In this kernel graph, s and t lie on the boundary of the outer face f_∞ , and every inner face shares an edge with f_∞ (see Fig. 1a). Routing then proceeds counterclockwise along f_∞ from s to t , switching to a clockwise traversal of an inner face upon encountering a failure (Fig. 1b).

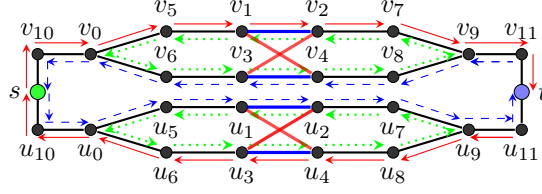
Compared to basic routing, source-matched routing incurs higher computational and memory costs, as it requires computing and storing routing tables for every pair of nodes, i.e., $\Theta(n^2)$ tables, whereas basic (only destination-based) routing needs only $\Theta(n)$ tables.

Dynamic failures transform the overlay graph from a static to a dynamic structure, rendering some classical failover routing rules ineffective.

As a common technique, the *bouncing reroute rule*, illustrated in Fig. 1b, upon a failed arc, bounces a packet back via its reverse, detours around the induced cycle, and reencounters the failed edge in reverse to terminate the detour. This technique is widely used [5, 8, 17].

However, as shown in Fig. 1b, if the dynamic failure becomes recovered on the second visit, the packet falls into a routing loop within the current induced cycle. A similar issue has also been observed in $(k-1)$ -resilient routing against dynamic failures on k -edge-connected graphs using three-bit rewriting [10].

Moreover, in Theorem 2, we show that the 2-resilient source-matched routing by Dai et al. [8, Algorithm 1] no longer functions correctly even for two semi-dynamic failures. Due to space limitations, we present in Fig. 2 a counterexample that sketches the proof of Theorem 2.



■ **Figure 2** Example of applying the 2-resilient source-matched routing algorithm proposed by Dai et al. [8, Algorithm 1] to a graph $G = (V, E)$ shown as bold lines without arrows in Fig. 2 ([8, Fig. 1]) for the source-destination pair (s, t) to obtain its kernel graph \mathcal{G} by excluding these four red bold lines: $\{\{v_1, v_4\}, \{v_2, v_3\}, \{u_1, u_4\}, \{u_2, u_3\}\}$ as shown in [8, Fig. 2], where a kernel graph \mathcal{G} is a subgraph of G , s.t., for any two failures $F \subseteq E$, if $s - t$ is connected in $G \setminus F$ then $s - t$ is also connected in $\mathcal{G} \setminus F$. By [8, Definition 6.2], a forwarding scheme $\Pi^{(s,t)}$ defines a link-circular forwarding function at each node of \mathcal{G} , and we can easily verify that $\Pi^{(s,t)}$ is 2-resilient against static failures. In this figure, $\Pi^{(s,t)}$ is illustrated by solid (red) arcs, dotted (green) arcs, and dashed (blue) arcs respectively, s.t., at a node v , a packet from an incoming arc (u, v) is forwarded to an outgoing arc (v, w) that has the same dash pattern (color) as (u, v) . If an outgoing arc (v, w) is failed, then the arc (w, v) is treated as an incoming arc to continue forwarding on the dash pattern (color) of (w, v) , while a packet originated at s can select either the solid (red) arc (s, v_{10}) or the dashed (blue) arc (s, u_{10}) arbitrarily to start. However, this forwarding scheme $\Pi^{(s,t)}$ is not 2-resilient against semi-dynamic failures. For semi-dynamic failures $F = \{\{v_1, v_2\}, \{v_7, v_9\}\}$, by starting at s and following forwarding rules (red arcs), the packet goes through $(s, v_{10}, v_0, v_5, v_1, v_2, v_7)$ to meet the first failure (v_7, v_9) , and then it is rerouted by the dashed forwarding rules (green arcs) to traverse (v_7, v_2) to hit the second failure (v_2, v_1) . Now, $\Pi^{(s,t)}$ makes the packet stuck in the connected component on $\{v_2, v_7\}$, but in the graph $G \setminus F$, there is still a path from v_7 to t , e.g., $(v_7, v_2, v_3, v_4, v_8, v_9, v_{11}, t)$, implying that $\Pi^{(s,t)}$ is not 2-resilient against semi-dynamic failures. Moreover, after adapting $\Pi^{(s,t)}$ by additionally enforcing clockwise link-circular routing at v_1 and v_2 to include $\{\{v_1, v_4\}, \{v_2, v_3\}\}$, we can easily verify that it becomes a 2-resilient source-matched routing against semi-dynamic failures.

► **Theorem 2.** *There exists a graph G where the 2-resilient source-matched routing of Dai et al. [8, Algorithm 1], designed for static failures, fails under two semi-dynamic failures, even though G admits such a routing scheme.*

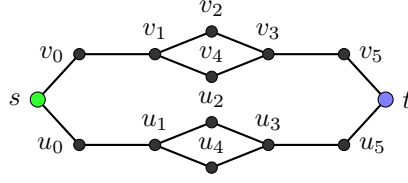
3.2 Impossibility of Handling Dynamic Failures in General Graphs

After discussing the limitations of existing failover approaches under dynamic failures, we show in Theorem 3 that no source-matched routing scheme can tolerate two dynamic failures. The proof idea is illustrated in Fig. 3, while the formal proof is deferred to Appendix A.

► **Theorem 3.** *There exists a 2-edge-connected planar graph G as shown in Fig. 3, where it is impossible to have a 2-resilient source-matched routing scheme against dynamic failures without rewriting bits in packet headers.*

For the counter-example graph G as shown in Fig. 3, fixing $\pi_s(\perp) = v_0$ when $F_s = \emptyset$, the routing functions at v_1 and v_3 cannot know whether an incoming packet currently should either continue searching a path towards t in $G[V' \cup \{s, t\}]$ or finding a path back to s in $G[V' \cup \{s, t\}]$ to try paths in $G[U' \cup \{s, t\}]$. Simply, by rewriting one bit in packet headers, the source-matched routing functions can resolve this weakness to achieve 2-resilience against dynamic failures again in G . Now, a fundamental question arises: Can we achieve k -resilience against dynamic failures in a general graph by only rewriting $O(\log k)$ bits in packet headers?

In light of Theorem 4, we demonstrate that achieving perfect resilience through the modification of $O(\log k)$ bits is impossible and defer its formal proof to Appendix A.



■ **Figure 3** Counter-example topology G for 2-resilient source-matched routing scheme against dynamic failures, where s is the source and t is the destination. Let $V' = \{v_0, \dots, v_5\}$ and $U' = \{u_0, \dots, u_5\}$. By symmetry, w.l.o.g., we can assume $\pi_s(\perp) = v_0$ when $F_s = \emptyset$. Then, we can show that each node $v \in V' \cup \{s\}$ must use a link-circular routing function, which has only two possible orderings for its neighbors, i.e., clockwise and counter-clockwise for the shown drawing. For instance, the clockwise and counter-clockwise orderings for v_1 are $\langle v_0, v_2, v_4 \rangle$ and $\langle v_0, v_4, v_2 \rangle$, respectively. We can further show that v_1 and v_3 must have the same type of orderings (clockwise or counter-clockwise), otherwise a routing loop can occur, e.g., if v_1 and v_3 select clockwise and counter-clockwise orderings respective, then a loop $(s, v_0, v_1, v_2, v_3, v_4, v_1, v_0, s)$ occurs for a static failure $F = \{s, u_0\}$. When v_1 and v_3 both use the clockwise (resp., counter-clockwise) ordering, for a dynamic failure $\{v_2, v_3\} \in F$ (resp., $\{v_3, v_4\} \in F$), let (v_2, v_3) (resp., (v_4, v_3)) be down and (v_3, v_2) (resp., (v_3, v_4)) be up. Then, a routing loop: $(s, v_0, v_1, v_2, v_1, v_4, v_3, v_2, v_1)$ (resp., $(s, v_0, v_1, v_4, v_1, v_2, v_3, v_4, v_1)$) appears and the packet originated at s cannot reach t even there is an $s - t$ path containing no dynamic failure. A similar proof can be given when $\pi_s(\perp) = u_0$ for $F_s = \emptyset$.

► **Theorem 4.** *There exists graphs for which any resilient source-matched routing that can tolerate $2k$ dynamic failures needs rewriting of at least k bits in packet headers for $k \in \mathbb{N}$.*

4 2-Resilient Routing Against Dynamic Failures on Planar Graphs

In contrast to our impossibility result, we now present algorithms achieving 2-resilience against dynamic failures on planar graphs. We first reduce the general problem, then introduce the elementary routing rules based on ear-decomposition and faces, and finally describe our 2-resilient basic routing scheme via one- or two-bit rewriting.

4.1 Problem Reductions and Basic Observations

By Claim 5, we reduce the problem to the case where G is a *simple* 2-edge-connected plane graph, and in Lemma 6 we highlight an important observation on this simplified structure.

▷ **Claim 5 (Problem Reduction).** The problem of devising a 2-resilient routing algorithm against dynamic failures in a general planar graph $G = (V, E)$ can be reduced to the case of 2-edge-connected planar graphs. Throughout, we can assume that G is a simple (plane) graph, i.e., it contains no parallel edges and has a plane embedding already.

Proof. For a ≥ 3 -edge-connected graph G , Dai et al. [10] showed that a 2-resilient routing scheme against dynamic failures can be devised using three arc-disjoint arborescences, building on the original idea of [5]. Alternatively, we can obtain a 2-edge-connected graph G' of by replacing any edge $\{u, v\}$ in G with two edges $\{u, \nu_{u,v}\}$ and $\{\nu_{u,v}, v\}$, s.t., any 2-resilient routing scheme for G' can also work for G .

Let $G = (V, E)$ be a 1-edge-connected graph. Define the set of *bridges* as $E' := \{e \in E \mid G \setminus \{e\} \text{ is disconnected}\}$. Removing all bridges yields a collection \mathcal{C} of connected components of $G \setminus E'$. By contracting each component $C_i \in \mathcal{C}$ to a single node v_{C_i} , we obtain an abstract tree $\mathcal{T}(G) := (\{v_{C_i} : C_i \in G \setminus E'\}, E')$, where every edge $e \in E'$ connects the two components in $G \setminus E'$ that e originally joined.

Since $\mathcal{T}(G)$ is a tree, one can directly construct a 2-resilient routing on $\mathcal{T}(G)$. Moreover, for each component C_i , which is itself a subgraph of G with edge-connectivity at least 2, if we can compute a 2-resilient routing within C_i that reaches the endpoint corresponding to the bridge edge in E' , then combining these local routings with the tree routing on $\mathcal{T}(G)$ yields a global 2-resilient routing for G .

If G contains parallel edges between two nodes $u, v \in V$, each such edge $\{u, v\}^i$ can be replaced by a path of length two, namely $\{u, \mu_{uv}^i\}$ and $\{\mu_{uv}^i, v\}$, where μ_{uv}^i is a newly introduced auxiliary node. \blacktriangleleft

► **Lemma 6.** *If $G = (V, E)$ is a ≥ 2 -edge-connected plane (embedded planar) graph, then each edge $e \in E$ is shared by two distinct faces, i.e., $e \in E(f_i)$ and $e \in E(f_j)$, where $f_i, f_j \in \mathcal{F}$ (\mathcal{F} including the outer face) and $i \neq j$.*

Proof. If G is ≥ 2 -edge-connected, then every edge $e \in E$ lies on a cycle. By [12, Lema 4.2.2], each edge $e \in E$ on a cycle of G must be shared by two distinct faces. \blacktriangleleft

► **Remark 7.** Even in the reduced setting, the problem remains challenging. Basic routing based solely on faces is already non-trivial: two failures on a face f_i may require traversing its boundary in both clockwise and counterclockwise directions. This necessitates at least one rewritable bit to indicate the current direction, since an arc may be used in opposite directions by two adjacent faces sharing the same edge. This observation motivates us to design routing schemes that exploit not only faces but also additional structural information.

4.2 Routing on Ear Decomposition and Faces

Our 2-resilient routing algorithms combine ear-based routing with face detours. We begin by introducing the basic routing based on ear-decomposition and faces in the following.

4.2.1 Routing Rules Based on Ear Decomposition.

Several variants of *ear-decomposition* appear in the literature (e.g., [19, 21, 23, 26]). In this work, we adopt the definition provided in the textbook [33].

Ear Decomposition. An *ear decomposition* $\mathcal{P}(G)$ (abbreviated, \mathcal{P}) of a graph G is a sequence of subgraphs $G = P_0 \cup P_1 \cup \dots \cup P_k$, where P_0 is a cycle in G , and for each $i \geq 0$, P_{i+1} is a path whose endpoints lie in $P_0 \cup P_1 \cup \dots \cup P_i$, while all internal vertices of P_{i+1} are disjoint from $P_0 \cup P_1 \cup \dots \cup P_i$. The cycle P_0 is called the *initial ear*, and each P_{i+1} (for $i \geq 0$) is called an *ear*. If the two endpoints of P_{i+1} coincide (so that P_{i+1} forms a cycle), it is referred to as a *closed ear*; otherwise, it is an *open ear* [19, 23, 25, 33]. In particular, the family $\{E(P_i)\}_{i=0}^k$ forms a partition of $E(G)$; that is, every edge $e \in E(G)$ lies in exactly one ear $P_i \in \mathcal{P}$. Robbins (1939) [25, 33] showed that a graph G is 2-edge-connected if and only if it admits an ear-decomposition, in which each P_i for $i \geq 1$ is either an open ear or a closed ear, and every cycle in a 2-edge-connected graph G can be the *initial ear* P_0 in some such decomposition. More specifically, one can compute an ear decomposition such that a specified vertex $v \in V$ is contained in the initial ear [19, 21, 23, 26, 28]. *In the following, we assume that the destination $t \in V$ is implicitly contained in the initial ear.*

► **Definition 8** (Routing Rules on Ears). *For a closed ear $P_i \in \mathcal{P}$ (including the initial ear), P_i itself is a cycle. For an open ear $P_i \in \mathcal{P}$ with endpoints $u_{P_i}, v_{P_i} \in V(P_i)$ and $i \geq 1$, we form the auxiliary cycle $P_i \cup \{u_{P_i}, v_{P_i}\}$ by adding the edge $\{u_{P_i}, v_{P_i}\}$.*

For each (either open or closed) ear $P_i \in \mathcal{P}$, we define the left (resp., right) routing on P_i (excluding the added edge $\{u_{P_i}, v_{P_i}\}$ if P_i is open) as the traversal of P_i in the direction such that the interior of the corresponding cycle— P_i itself if closed, or $P_i \cup \{u_{P_i}, v_{P_i}\}$ if open—lies on the left-hand (resp., right-hand) side during your traversing. In the case of an open ear, the endpoint visited last under the left (resp., right) routing is called the left (resp., right) endpoint. For a closed ear, the left and right endpoints coincide, yielding a unique endpoint.

Finally, for each ear $P_i \in \mathcal{P}$, we define \vec{P}_i (resp., \overleftarrow{P}_i) as the directed cycle (resp., path, if P_i is open) obtained by traversing P_i in the right (resp., left) direction, from its left (resp., right) endpoint to its right (resp., left) endpoint.

Since each edge $\{u, v\} \in E$ uniquely belongs to some ear $P_i \in \mathcal{P}$, each oriented arc of $\{u, v\}$ is uniquely contained in either \vec{P}_i or \overleftarrow{P}_i , even if P_i is closed.

By Definition 9, we define a function that maps each node to an ear to start routing.

► **Definition 9.** Since a node $v \in V$ may appear in multiple ears of \mathcal{P} , we define the function $\mathcal{P}_{\min}(v)$ that returns the earliest ear $P_i \in \mathcal{P}$ (in terms of index i) containing $v \in V \setminus \{t\}$ as an internal node in the subgraph $P_0 \cup P_1 \cup \dots \cup P_i \subseteq G$. In particular, let $\mathcal{P}_{\min}(t) = P_0$.

4.2.2 Routing Rules on Faces of Plane Graph

After defining the ear-based routing rules, Definition 10 presents the face-based routing rules.

► **Definition 10 (Routing Rules on Faces).** Given a plane embedding of a planar graph G , for each inner face $f_i \in \mathcal{F}$ of G , which is bounded by a cycle $C(f_i) \subseteq G$, we define the left (resp., right) routing on f_i as the traversal on the boundary (cycle) $C(f_i)$ of f_i in the direction such that the bounded region of $C(f_i)$ lies on the left-hand side (resp., right-hand side) during your traversing. Analogously, for the outer face $f_\infty \in \mathcal{F}$ of G , the left (resp., right) routing on f_∞ is defined as the traversal of the cycle $C(f_\infty)$ in the direction such that the unbounded region of $C(f_\infty)$ lies to the left-hand side (resp., right-hand side) of the traverser.

Henceforth, for each face $f_i \in \mathcal{F}$ (including f_∞), let \overleftarrow{f}_i (resp., \vec{f}_i) denote the directed cycle obtained by traversing the boundary $C(f_i)$ of f_i according to the left (resp., right) routing of f_i . And we define a collection of directed cycles $\vec{\mathcal{F}} := \bigcup_{f_i \in \mathcal{F}} \vec{f}_i$, (resp., $\overleftarrow{\mathcal{F}} := \bigcup_{f_i \in \mathcal{F}} \overleftarrow{f}_i$).

By Lemma 11, the face-based routing defined in Definition 10 yields consistent routing rules.

► **Lemma 11.** Given a set of faces \mathcal{F} of a plane graph G , fix a routing direction (either left or right) for all faces. Then no two directed cycles in $\vec{\mathcal{F}}$ (resp., $\overleftarrow{\mathcal{F}}$) share an arc. Moreover, once the routing direction is fixed (either $\vec{\mathcal{F}}$ or $\overleftarrow{\mathcal{F}}$), each orientation of an edge $\{u, v\} \in E$, i.e., an arc (u, v) or (v, u) , uniquely determines the corresponding face $f_i \in \mathcal{F}$. In particular, for any edge $\{u, v\} \in E$, if $(u, v) \in \vec{f}_i$ and $(v, u) \in \overleftarrow{f}_j$ with $\vec{f}_i \in \vec{\mathcal{F}}$ and $\overleftarrow{f}_j \in \overleftarrow{\mathcal{F}}$, then $f_i = f_j$.

Proof. We present the argument for right routing; the case of left routing follows analogously. By Lemma 6, every edge $e \in E$ is incident with exactly two distinct faces, say $f_i, f_j \in \mathcal{F}$.

Consider an embedding of G on the Euclidean plane, and let $\{u, v\} \in E$ be an edge shared by f_i and f_j . The infinite line by expanding $\{u, v\}$ separates the plane into two half-planes, with f_i lying entirely on one side and f_j entirely on the other. Now, when traversing an arc (u, v) (resp., (v, u)), exactly one of these half-planes (and thus exactly one of the two faces) lies to the right of the traversal. Consequently, in a right routing, the directed cycles \vec{f}_i and \vec{f}_j cannot share the same directed arc of $\{u, v\}$. Therefore, once the routing direction

is fixed, each arc (u, v) of an edge $\{u, v\} \in E$ is contained in exactly one directed cycle \vec{f}_i , which uniquely determines the corresponding face $f_i \in \mathcal{F}$.

By definition, for any edge $\{u, v\}$ and any face $f_i \in \mathcal{F}$, if the arc (u, v) belongs to \vec{f}_i , then the reverse arc (v, u) belongs to \overleftarrow{f}_i . \blacktriangleleft

In Definition 12, we define the rightmost and leftmost nodes on each face respectively, which serve as checkpoints to terminate face-based routing.

► Definition 12 (Associated Ear and Leftmost/Rightmost Node of a Face). *Given a 2-edge-connected plane graph $G = (V, E)$, for each face $f_i \in \mathcal{F}$ (including the outer face f_∞), the associated ear $\mathcal{P}_{\text{ass}}(f_i)$ of f_i is the first ear $P_j \in \mathcal{P}$ (with minimum index j) such that $E(f_i) \cap E(P_j) \neq \emptyset$. Given the associated ear P_j of f_i , let $e^* \in E(P_j) \cap E(f_i)$ be the edge traversed last along the right (resp., left) routing \vec{P}_j (resp., \overleftarrow{P}_j) in terms of the edges in $E(P_j) \cap E(f_i)$. The endpoint $v \in V(e^*)$ visited later in \vec{P}_j (resp., \overleftarrow{P}_j) is called the rightmost node (resp., leftmost node) of f_i , denoted by $v^r(f_i)$ (resp., $v^l(f_i)$). The leftmost or rightmost node of a face f_i is uniquely determined and independent of the traversal direction on faces. We have $v^l(f_i) = v^r(f_i)$ if they coincide as the unique endpoint of a closed ear P_j .*

Lemma 13 shows that our face-based routing can visit all nodes of a face $f_i \in \mathcal{F}$, including the leftmost and rightmost nodes of f_i , provided there is at most one failure in f_i .

► Lemma 13. *Given $\vec{\mathcal{F}}$ (resp., $\overleftarrow{\mathcal{F}}$), for a dynamic failure $\{u, v\} \in F$, if one of its arcs, say (v, u) , lies on a directed cycle $\vec{f}_i \in \vec{\mathcal{F}}$ (resp., $\overleftarrow{f}_i \in \overleftarrow{\mathcal{F}}$). Then, starting at u and traversing along \vec{f}_i (resp., \overleftarrow{f}_i), one can visit all nodes of $V(f_i)$ and eventually reach v , provided that the face f_i contains at most one failed edge, i.e., $|F \cap E(f_i)| \leq 1$.*

Proof. Our proof is carried out for the right routing $\vec{\mathcal{F}}$; the case of the left routing $\overleftarrow{\mathcal{F}}$ follows analogously. Suppose $(v, u) \in \vec{f}_i$ is failed. Starting from u , the outgoing link (arc) along \vec{f}_i is not failed, and hence there exists a directed sub-path from u to any node $t \in V(f_i)$ that is entirely contained in $\vec{f}_i \setminus \{(v, u)\}$ that is a directed path from u to v along \vec{f}_i . \blacktriangleleft

4.3 Dynamic 2-Resilient Basic Routing with One- or Two-Bit Rewriting

Now, we formally present dynamic 2-resilient routing schemes on planar graphs using one- or two-bit rewriting. We first describe the two-bit rewriting algorithm and then show how to refine it to a one-bit algorithm. Finally, we provide the runtime analysis of these algorithms.

4.3.1 2-Resilient Basic Routing Algorithm by Rewriting Two Bits

We first present our 2-resilient routing scheme on plane graphs by rewriting two bits, implemented in Algorithm 1 and Algorithm 2, with an illustrative example in Fig. 4.

The core idea of the two-bit rewriting algorithm is to route packets through ears $P_\ell \in \mathcal{P}$ in a waterfall manner towards $P_{\ell-1}$ when $\alpha = 0$: a packet first follows \vec{P}_ℓ and, upon encountering a failure, switches to \overleftarrow{P}_ℓ . If every ear contains at most one failure, the packet eventually reaches t without invoking face-based routing (This case is proved by Lemma 14).

If a second failure $(u, v) \in \overleftarrow{P}_\ell$ occurs on the same ear P_ℓ , routing switches to face-based mode ($\alpha = 1$) on the adjacent faces f_i and f_j , with $(v, u) \in \vec{f}_i$ and $(v, u) \in \overleftarrow{f}_j$.

The algorithm first attempts \vec{f}_i with $\beta = 0$, and if unsuccessful, retries \overleftarrow{f}_j with $\beta = 1$. Since at least one of $\{\vec{f}_i, \overleftarrow{f}_j\}$ contains a failure, the packet is guaranteed to reach the rightmost

■ **Algorithm 1** A 2-resilient basic routing scheme in 2-edge-connected planar graphs against dynamic failures, achieved by rewriting one or two bits in packet headers.

Data: a plane graph $G = (V, E)$ with ears \mathcal{P} and faces \mathcal{F} , and destination $t \in V$;

- 1 Let $v_{\text{cursor}} \in V$ denote the current node processing a packet;
- 2 For a packet originated at $v \in V$, let $v_{\text{cursor}} \leftarrow v$ and decide an outgoing link (arc) $(v_{\text{cursor}}, u) \in \overleftarrow{P_{\text{curr}}}$ (left routing) with the current ear $P_{\text{curr}} \leftarrow \mathcal{P}_{\min}(v_{\text{cursor}})$;
- 3 Initialize $\alpha \leftarrow 0$; // Bit $\alpha \in \{0,1\}$: 0 indicates routing on ears, otherwise on faces
- 4 Initialize $\beta \leftarrow 0$; // Bit $\beta \in \{0,1\}$: 0 means right routing on faces, otherwise left
- 5 **while** the packet is not delivered to $t \in V$ **do**
- 6 **if** $\alpha = 0$ **then**
- 7 /* Routing on Ears */
- 7 v_{cursor} decides the current ear $P_{\text{curr}} \in \mathcal{P}$ and its direction by an incoming link (arc), i.e., either $(u, v_{\text{cursor}}) \in \overleftarrow{P_{\text{curr}}}$ or $(u, v_{\text{cursor}}) \in \overrightarrow{P_{\text{curr}}}$;
- 8 **if** the current node v_{cursor} is not an endpoint of P_{curr} **then**
- 9 decides the outgoing link: $(v_{\text{cursor}}, w) \in \overleftarrow{P_{\text{curr}}}$ or $(v_{\text{cursor}}, w) \in \overrightarrow{P_{\text{curr}}}$;
- 10 **else**
- 11 update $P_{\text{curr}} \leftarrow \mathcal{P}_{\min}(v_{\text{cursor}})$ and follow $\overleftarrow{P_{\text{curr}}}$ (left) for outgoing link;
- 12 **if** the outgoing link (arc) (v_{cursor}, w) is failed **then**
- 13 **if** $(v_{\text{cursor}}, w) \in \overrightarrow{P_{\text{curr}}}$, **then** $\alpha \leftarrow 1$;
- 14 reverse (v_{cursor}, w) to use (w, v_{cursor}) as incoming link to continue;
- 15 **else**
- 16 moving from v_{cursor} to next node w ;
- 17 **else**
- 18 /* Routing on Faces; The bit β is not required by Algorithm 3 */
- 18 apply the subroutine of either Algorithm 2 (rewriting 2-bits) or Algorithm 3 (rewriting 1-bit) with the incoming link (w, v_{cursor}) as the input;
- 19 $\alpha \leftarrow 0$;

node of either $v^r(f_i)$ or $v^r(f_j)$ to exit P_ℓ , thereby resuming ear-based routing on some $P_{\ell'}$ with $\ell' < \ell$ to finally reach t . Note that P_ℓ need not coincide with $\mathcal{P}_{\min}(f_{i'})$ for $i' \in \{i, j\}$.

► **Lemma 14.** *If every ear in \mathcal{P} contains at most one failure in F , then Algorithm 1 routes a packet to the destination t under two dynamic failures by only traversing along ears.*

Proof. In Algorithm 1, when a packet begins at an internal node in an (including the initial) ear $P_{\text{curr}} \in \mathcal{P}$, it is first routed along the left direction $\overleftarrow{P_{\text{curr}}}$ of P_{curr} . Upon encountering a failure, the traversal switches to the right direction $\overrightarrow{P_{\text{curr}}}$. If each ear satisfies $|E(P_{\text{current}}) \cap F| \leq 1$, then one of the two routings, i.e., $\overleftarrow{P_{\text{curr}}}$ or $\overrightarrow{P_{\text{curr}}}$, always leads to an endpoint of P_{current} . From there, the packet proceeds into an earlier ear $P_i \in \mathcal{P}$ with $i < \text{curr}$, and ultimately reaches P_0 to arrive at t . Since ear indices strictly decrease, no ear is revisited, and thus routing loops cannot occur. ◀

By Lemma 15, if two failures $F = \{e_1, e_2\}$ are only shared by distinct faces $f_i, f_j \in \mathcal{F}$, then this scenario can be excluded from the resilient routing design, as the node, where a packet is currently staying, and destination t lie in separate connected components of $G \setminus F$.

Algorithm 2 Two-Bit Rewriting Subroutine of Algorithm 1.

```

1 repeat
2   Traverse the directed cycle (face)  $\vec{f}_i$  with  $(w, v_{\text{cursor}}) \in \vec{f}_i$  if  $\beta = 0$ , otherwise
   traverse along  $\overleftarrow{f}_i$  with  $(w, v_{\text{cursor}}) \in \overleftarrow{f}_i$ ;
3 until an outgoing link  $(v_{\text{cursor}}, w)$  in  $\vec{f}_i$  (resp.,  $\overleftarrow{f}_i$ ) is failed or  $v_{\text{cursor}} = v^r(f_i)$ ;
4 if arriving at the rightmost  $v^r(f_i)$  of  $f_i$ , i.e.,  $v_{\text{cursor}} = v^r(f_i)$  then
5   decide an outgoing link (arc)  $(v_{\text{cursor}}, w) \in \overrightarrow{P_{\text{curr}}}$  (right routing) with the current
   ear  $P_{\text{curr}} \leftarrow \mathcal{P}_{\min}(v_{\text{cursor}})$ , and then move to  $w$ , i.e.,  $v_{\text{cursor}} \leftarrow w$ ;
6 else
7    $\beta \leftarrow 1$ ;
8    $v_{\text{cursor}}$  updates the incoming link (arc) as  $(u, v_{\text{cursor}}) \in \overleftarrow{P_{\text{curr}}}$  (left routing) with
   the current ear  $P_{\text{curr}} \leftarrow \mathcal{P}_{\min}(v_{\text{cursor}})$ ; //  $\mathcal{P}_{\min}(v_{\text{cursor}}) \leq P_i$ , where the ear  $P_i$ 
   originally has two failures, triggering routing on faces

```

► **Lemma 15.** Let $G = (V, E)$ be a 2-edge-connected plane graph with face set \mathcal{F} . If two distinct faces $f_i, f_j \in \mathcal{F}$ share two distinct edges $e_1, e_2 \in E$, i.e., $e_1, e_2 \in E(f_i) \cap E(f_j)$, then $G \setminus \{e_1, e_2\}$ is disconnected.

Proof. Consider the plane dual graph $G^* = (U^*, X^*)$ of $G = (V, E)$, where each vertex $u_i^* \in U^*$ corresponds to a face $f_i \in \mathcal{F}$ of G , and each edge $e \in E$ shared by two faces $f_i, f_j \in \mathcal{F}$ corresponds to an edge $\{u_i^*, u_j^*\} \in X^*$. Note that G^* may contain multi-edges even if G is simple. Since e_1, e_2 are common to f_i and f_j , their duals e_1^*, e_2^* are parallel edges between u_i^* and u_j^* ; thus, $\{e_1^*, e_2^*\}$ forms a simple cycle of length two in G^* .

By the cut-cycle duality for plane graphs [12], a set $J \subseteq E$ is a *minimal edge cut (bond)* in G iff the dual set $X^*(J) := \{e^* \in X^* : e \in J\}$ is a simple cycle in G^* . Applying this to $J = \{e_1, e_2\}$ shows that $\{e_1, e_2\}$ is a bond of G . Thus, deleting e_1 and e_2 disconnects G . ◀

Now, we formally prove the correctness of the 2-bits rewriting algorithm in Theorem 16.

► **Theorem 16.** By combining Algorithm 1 with Algorithm 2, a packet can be successfully routed to its destination t under two dynamic failures by rewriting two bits in its header.

Proof. By Lemma 14, we may assume that both failures $e_1, e_2 \in F$ lie on the same ear $P_{\text{curr}} \in \mathcal{P}$, and that the packet is currently located at a node v_{cursor} between e_1 and e_2 on P_{curr} ; otherwise, t can be reached solely via ear-based routing.

Let $e_2 = (u, v)$ be the second failure, with $(u, v) \in \overrightarrow{P_{\text{curr}}}$. By Lemma 11, there exist two distinct faces $f_i, f_j \in \mathcal{F}$ (with one possibly being the outer face), such that $(v, u) \in \vec{f}_i$ and $(v, u) \in \overleftarrow{f}_j$. By Lemma 15, at least one of f_i, f_j contains only e_2 but not e_1 .

We first traverse \vec{f}_i starting at u due to $\beta = 0$. If no further failure occurs, then by Lemma 13, we reach the rightmost node $v^r(f_i)$ of f_i , closer to the right endpoint of P_{curr} than (u, v) along $\overrightarrow{P_{\text{curr}}}$. Note that the directed subpath of $\overrightarrow{P_{\text{curr}}}$ from $v^r(f_i)$ cannot include either e_1 or e_2 by Definition 12. From $v^r(f_i)$, either routing continues along $\overrightarrow{P_{\text{curr}}}$ to reach the endpoint of P_{curr} , or $v^r(f_i)$ is already an internal node of an ear P_j with $j < \text{curr}$, from which t is reachable by ear-based routing alone. Note that P_{curr} may not be $\mathcal{P}_{\min}(f_i)$.

If instead traversal of \vec{f}_i encounters e_1 , switching back to ear-based routing either leads to the endpoint of P_{curr} along $\overleftarrow{P_{\text{curr}}}$ or to revisiting (u, v) along $\overleftarrow{P_{\text{curr}}}$. In this case, face-based routing resumes via \overleftarrow{f}_j (since $\beta = 1$ and $(v, u) \in \overleftarrow{f}_j$), eventually reaching the rightmost $v^r(f_j)$

Algorithm 3 One-Bit Rewriting Subroutine of Algorithm 1.

```

1 repeat
2   Traverse the directed cycle (face)  $\vec{f}_i$  with  $(w, v_{\text{cursor}}) \in \vec{f}_i$ . Upon a failure
   (outgoing arc)  $(v_{\text{cursor}}, w) \in \vec{f}_i$ , update  $f_i \leftarrow f_j$ , s.t.,  $f_j \in \mathcal{F}$  and
    $(w, v_{\text{cursor}}) \in \vec{f}_j$ , and continue along  $\vec{f}_i$  ( $i = j$ ) ; // Only right routing on faces
3 until  $(v_{\text{cursor}}$  is an endpoint of  $P_\ell$ ) or  $(v_{\text{cursor}} = v^r(f_i) \wedge (u, v_{\text{cursor}}) \notin \vec{P}_\ell)$  or
    $(v_{\text{cursor}} = v^l(f_i) \wedge (u, v_{\text{cursor}}) \notin \overleftarrow{P}_\ell)$ , where  $P_\ell = \mathcal{P}_{\text{ass}}(f_i)$  and  $(u, v_{\text{cursor}}) \in \vec{f}_i$  is the
   incoming link;
4 decide an outgoing link (arc)  $(v_{\text{cursor}}, w) \in \overrightarrow{P_{\text{curr}}}$  if  $v_{\text{cursor}} = v^r(f_i)$ , otherwise
    $(v_{\text{cursor}}, w) \in \overleftarrow{P_{\text{curr}}}$ , where  $P_{\text{curr}} \leftarrow \mathcal{P}_{\text{min}}(v_{\text{cursor}})$ , and move to  $w$ , i.e.,  $v_{\text{cursor}} \leftarrow w$ ;

```

► **Theorem 18.** *Algorithm 1, in conjunction with Algorithm 3, can route a packet to the destination t under two dynamic failures by rewriting one bit in the packet header.*

Proof. It remains to consider the case where a packet gets stuck on an ear $P_{\text{curr}} \in \mathcal{P}$ containing two failures, where first hitting $(u^1, v^1) \in \overleftarrow{P_{\text{curr}}}$ and then encountering $(u^2, v^2) \in \overrightarrow{P_{\text{curr}}}$.

Let $f_i \in \mathcal{F}$ with associated ear $P_\ell := \mathcal{P}_{\text{ass}}(f_i)$ and $(v^2, u^2) \in \vec{f}_i$. Then \vec{f}_i is consistent with $\overleftarrow{P_{\text{curr}}}$ (Lemma 17), while its consistency with P_ℓ depends on whether $P_{\text{curr}} = P_\ell$.

If f_i contains only one failure, we will show a traversal along \vec{f}_i from u^2 terminates at either the rightmost node $v^r(f_i)$, the leftmost node $v^l(f_i)$, or an endpoint of P_ℓ (Algorithm 3, Lines 1–3;), from which ear-based routing resumes to reach t . By Lemma 13, all nodes of $V(f_i)$ are eventually visited. If the current node v_{cursor} is an endpoint of P_ℓ , the traversal along \vec{f}_i terminates and routing leaves P_ℓ through this endpoint. We now assume $v^r(f_i) \neq v^l(f_i)$; otherwise, $v^r(f_i) = v^l(f_i)$ is itself an endpoint of P_ℓ (Definition 12).

If $P_{\text{curr}} = P_\ell$, then \vec{f}_i is consistent with $\overleftarrow{P_\ell}$. In this case, we can show the traversal must halt at the rightmost node $v^r(f_i)$ (not at $v^l(f_i)$). Let $(u, v^r(f_i)) \in \vec{f}_i$ be the arc preceding $v^r(f_i)$ along \vec{f}_i (we may have $u^2 = v^r(f_i)$ with $v^2 = u$). Suppose $\{u, v^r(f_i)\} \in E(P_\ell)$. Then, by $(u, v^r(f_i)) \in \vec{f}_i$ and Lemma 17, it follows that $(u, v^r(f_i)) \in \overleftarrow{P_\ell}$. By Definition 10, this would imply $(v^r(f_i), u) \in \overrightarrow{P_\ell}$, contradicting that $v^r(f_i)$ is the rightmost node of f_i . Hence, it implies $\{u, v^r(f_i)\} \notin E(P_\ell)$ and the traversal along \vec{f}_i halts at $v^r(f_i)$. To show that the traversal cannot halt at $v^l(f_i)$, assume there exists an arc $(\mu, v^l(f_i)) \in \vec{f}_i$. If $(\mu, v^l(f_i)) \notin \overleftarrow{P_\ell}$, then by Lemma 17, we have $\{\mu, v^l(f_i)\} \notin E(P_\ell)$. By Definition 12, $v^l(f_i)$ must have a neighbor ν with $(v^l(f_i), \nu) \in \vec{f}_i$ and $(v^l(f_i), \nu) \in \overleftarrow{P_\ell}$. This implies ν is visited after $v^l(f_i)$ along $\overleftarrow{P_\ell}$, contradicting that $v^l(f_i)$ is the leftmost node. Hence, $(\mu, v^l(f_i)) \in \vec{f}_i$ implies $(\mu, v^l(f_i)) \in \overleftarrow{P_\ell}$, indicating that the traversal cannot stop at $v^l(f_i)$. Consequently, from $v^r(f_i)$ the ear-based traversal along $\overrightarrow{P_{\text{curr}}}$ leaves P_{curr} without further failures and finally reaches t .

If $P_{\text{curr}} \neq P_\ell$, then \vec{f}_i can be consistent with either $\overleftarrow{P_\ell}$ or $\overrightarrow{P_\ell}$. Starting at $u^2 \notin V(P_\ell)$, if \vec{f}_i consistent with $\overleftarrow{P_\ell}$ (resp., $\overrightarrow{P_\ell}$), the traversal along \vec{f}_i stops at $v^r(f_i)$ (resp., $v^l(f_i)$) since the incoming link (arc) $(w, v_{\text{cursor}}) \notin \overleftarrow{P_\ell}$ (resp., $(w, v_{\text{cursor}}) \notin \overrightarrow{P_\ell}$). As P_ℓ contains no failure ($\ell < \text{curr}$), the ear-based routing from $v^r(f_i)$ (resp., $v^l(f_i)$) along $\overrightarrow{P_\ell}$ (resp., $\overleftarrow{P_\ell}$) always leaves P_ℓ and continue traversing on the ears P_o with $o < \text{curr}$ to reach t .

If there are two failures on P_{curr} , upon hitting $(u^1, v^1) \in \overleftarrow{P_{\text{curr}}}$ (stopping at u^1), by Lemma 17, we know $(v^2, u^2) \in \vec{f}_i$ and $(u^1, v^1) \in \vec{f}_i$ since \vec{f}_i is consistent with $\overleftarrow{P_{\text{curr}}}$. In this case there exists a face $f_j \in \mathcal{F}$ with associated ear $P_{\ell'} := \mathcal{P}_{\text{ass}}(f_j)$ such that $(v^1, u^1) \in \vec{f}_j$ (By Lemma 15, only one failure on f_j), because \vec{f}_j is consistent with $\overrightarrow{P_{\text{curr}}}$. If $P_{\text{curr}} = P_{\ell'}$,

then traversing \vec{f}_j from u^1 halts at either an endpoint of $P_{\ell'}$ or at $v^l(f_j)$. Then, following $\overleftarrow{P_{\ell'}}$ from $v^l(f_j)$ (resp., the endpoint of $P_{\ell'}$) by ear-based routing can leave $P_{\ell'}$ to eventually reach t . However, if $P_{\text{curr}} \neq P_{\ell'}$, we can argue similarly as above, since no failures on $P_{\ell'}$. ◀

Complexity Analysis. We compute the routing rules of Algorithm 1 in a centralized model. The one- and two-bit rewriting approaches have identical time costs. For a graph G with n nodes and m edges, a planar embedding can be computed in $O(n)$ [2,20], an ear-decomposition in $O(n + m)$ [26], and ear-/face-based routing rules in $O(m)$. In the worst case, recomputing ear-decompositions per destination yields $O(n^2 + mn)$ total time. By comparison, source-matched routing [8] needs $O(nm)$ per source–destination pair, i.e., at least n^2 times slower. Memory usage is reduced from n^2 tables (source–destination) to only n (per destination). Moreover, our reliance on ear-decomposition and planar embeddings—well studied in parallel and distributed computing—underscores the practical feasibility of our routing schemes.

5 Conclusions and Future Work

In this paper, we investigated the resilience achievable by basic routing augmented with packet-header rewriting under dynamic failures. We proved that $k \geq 2$ resilience without any rewriting is impossible in planar graphs, and that, in general graphs, the perfect k -resilience is unfeasible by only rewriting $\log k$ bits. For planar graphs, we presented tight 2-resilient basic routing algorithms that require rewriting only one or two bits. Future work includes designing 2-resilient basic routing algorithms for general graphs with limited bit rewriting.

References

- 1 Mohammad Alizadeh et al. Data center TCP (DCTCP). In *SIGCOMM*. ACM, 2010. doi:10.1145/1851182.1851192.
- 2 John Boyer and Wendy Myrvold. On the cutting edge: Simplified $O(n)$ planarity by edge addition. *Journal of Graph Algorithms and Applications*, 8(3):241–273, January 2004. doi:10.7155/jgaa.00091.
- 3 Marco Chiesa et al. On the resiliency of randomized routing against multiple edge failures. In *ICALP*, 2016. doi:10.4230/LIPIcs.ICALP.2016.134.
- 4 Marco Chiesa et al. The quest for resilient (static) forwarding tables. In *INFOCOM*. IEEE, 2016. doi:10.1109/INFOCOM.2016.7524552.
- 5 Marco Chiesa et al. On the resiliency of static forwarding tables. *IEEE/ACM Trans. Netw.*, 25(2):1133–1146, 2017. doi:10.1109/TNET.2016.2619398.
- 6 Marco Chiesa et al. Fast reroute on programmable switches. *IEEE/ACM Trans. Netw.*, 29(2):637–650, 2021. doi:10.1109/TNET.2020.3045293.
- 7 Marco Chiesa et al. A survey of fast-recovery mechanisms in packet-switched networks. *IEEE Commun. Surv. Tutorials*, 23(2):1253–1301, 2021. doi:10.1109/COMST.2021.3063980.
- 8 Wenkai Dai et al. A tight characterization of fast failover routing: Resiliency to two link failures is possible. In *SPAA*, pages 153–163. ACM, 2023. doi:10.1145/3558481.3591080.
- 9 Wenkai Dai et al. On the resilience of fast failover routing against dynamic link failures, 2024. arXiv:2410.02021, doi:10.48550/arXiv.2410.02021.
- 10 Wenkai Dai et al. On the resilience of fast failover routing against dynamic link failures. In *IFIP Networking*, 2025.
- 11 Fabien de Montgolfier et al. Treewidth and hyperbolicity of the internet. In *NCA*, pages 25–32. IEEE Computer Society, 2011. doi:10.1109/NCA.2011.11.
- 12 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

- 13 Joan Feigenbaum et al. Brief announcement: On the resilience of routing tables. In *PODC*. ACM, 2012. doi:10.1145/2332432.2332478.
- 14 Klaus-Tycho Foerster et al. CASA: congestion and stretch aware static fast rerouting. In *INFOCOM*. IEEE, 2019. doi:10.1109/INFOCOM.2019.8737438.
- 15 Klaus-Tycho Foerster et al. Brief announcement: What can(not) be perfectly rerouted locally. In *DISC*, pages 46:1–46:3, 2020. doi:10.4230/LIPIcs.DISC.2020.46.
- 16 Klaus-Tycho Foerster et al. Grafting arborescences for extra resilience of fast rerouting schemes. In *INFOCOM*, pages 1–10. IEEE, 2021. doi:10.1109/INFOCOM42981.2021.9488782.
- 17 Klaus-Tycho Foerster et al. On the feasibility of perfect resilience with local fast failover. In *Symposium on Algorithmic Principles of Computer Systems (APOCS)*, 2021. doi:10.1137/1.9781611976489.5.
- 18 Phillipa Gill et al. Understanding network failures in data centers: measurement, analysis, and implications. In *SIGCOMM*. ACM, 2011. doi:10.1145/2018436.2018477.
- 19 S. Hannenhalli et al. A distributed algorithm for ear decomposition. In *Proceedings of ICCI'93: 5th International Conference on Computing and Information*, pages 180–184, 1993. doi:10.1109/ICCI.1993.315382.
- 20 John Hopcroft and Robert Tarjan. Efficient planarity testing. *J. ACM*, 21(4):549–568, October 1974. doi:10.1145/321850.321852.
- 21 Louis Ibarra and Dana Richards. Efficient parallel graph algorithms based on open ear decomposition. *Parallel Computing*, 19(8):873–886, 1993. doi:10.1016/0167-8191(93)90071-R.
- 22 Junda Liu et al. Ensuring connectivity via data plane mechanisms. In *NSDI*, 2013. URL: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/liu_junda.
- 23 L. Lovasz. Computing ears and branchings in parallel. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 464–467, 1985. doi:10.1109/SFCS.1985.16.
- 24 Athina Markopoulou et al. Characterization of failures in an operational ip backbone network. *IEEE/ACM transactions on Networking*, 16(4):749–762, 2008. doi:10.1145/1453698.1453699.
- 25 H. E. Robbins. A theorem on graphs, with an application to a problem of traffic control. *The American Mathematical Monthly*, 46(5):281–283, 1939. URL: <https://doi.org/10.2307/2303897>.
- 26 Jens M. Schmidt. A simple test on 2-vertex- and 2-edge-connectivity. *Information Processing Letters*, 113(7):241–244, 2013. doi:10.1016/j.ipl.2013.01.016.
- 27 Aman Shaikh et al. A case study of ospf behavior in a large enterprise network. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement*, IMW '02, pages 217–230. ACM, 2002. doi:10.1145/637201.637236.
- 28 Y. H. Tsin. On finding an ear decomposition of an undirected graph distributively. *Information Processing Letters*, 91(3):147–153, 2004. doi:10.1016/j.ipl.2004.04.004.
- 29 Daniel Turner et al. California fault lines: understanding the causes and impact of network failures. In *SIGCOMM*. ACM, 2010. doi:10.1145/1851182.1851220.
- 30 Balajee Vamanan et al. Deadline-aware datacenter tcp (D2TCP). In *SIGCOMM*. ACM, 2012. doi:10.1145/2342356.2342388.
- 31 Erik van den Akker and Klaus-Tycho Foerster. Short paper: Towards 2-resilient local failover in destination-based routing. In *ALGOCLOUD*. Springer, 2024. doi:10.1007/978-3-031-94677-6_2.
- 32 David Watson et al. Experiences with monitoring ospf on a regional service provider network. In *Proceedings of the 23rd International Conference on Distributed Computing Systems*, ICDCS '03, page 204, USA, 2003. IEEE Computer Society. doi:10.1109/ICDCS.2003.1203467.
- 33 Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 2 edition, September 2000.
- 34 Baohua Yang et al. Keep forwarding: Towards k-link failure resilient routing. In *INFOCOM*. IEEE, 2014. doi:10.1109/INFOCOM.2014.6848098.
- 35 David Zats et al. Detail: reducing the flow completion time tail in datacenter networks. In *SIGCOMM*. ACM, 2012. doi:10.1145/2342356.2342390.

A

 Deferred Proofs of Theorems and Lemmas in Section 3

We restate the theorems and lemmas before the proofs for ease of readability.

► **Theorem 3.** *There exists a 2-edge-connected planar graph G as shown in Fig. 3, where it is impossible to have a 2-resilient source-matched routing scheme against dynamic failures without rewriting bits in packet headers.*

Proof of Theorem 3. We first assume that a 2-resilient source-matched forwarding scheme $\Pi^{(s,t)}$ exists in the graph G as shown in Fig. 3. Then, for contradiction, we show that a packet originated at s cannot be routed to the destination t anymore, but is forwarded in a loop when there are two dynamic failures F in G , even if there exists an $s - t$ path in the graph $G \setminus F$.

Let $V' = \{v_0, \dots, v_5\}$ and $U' = \{u_0, \dots, u_5\}$. For each node $v \in V(G)$, we define a forwarding function $\pi_v(u, F_v)$ at v , where $F_v \subseteq F$ denotes a subset of dynamic failures F that incidents on v and the source-destination (s, t) is used implicitly in this proof. Clearly, the induced graphs $G[U']$ and $G[V']$ are symmetric. By symmetry, when $F_s = \emptyset$, an arbitrary node in $\{v_0, u_0\}$ can be chosen as the outgoing port for the packet originated at s . W.l.o.g., we assume that v_0 is chosen, i.e., $\pi_s(\perp) = v_0$ for $F_s = \emptyset$.

Let dynamic failures $F \subseteq E(G)$ be a set of arbitrary links, s.t., $|F| \leq 2$ and F can be empty. Next, we claim that, given $\pi_s(\perp) = v_0$ with $F_s = \emptyset$, for each node $v \in V' \cup \{s\}$, its routing function must be *link-circular* even when F are static failures. If $v \in V' \cup \{s\}$ has $\Delta_{G \setminus F}(v) = 1$, then $\pi_v(u, F_v) = u$, where $\pi_v \in \Pi^{(s,t)}$ and $u \in E_{G \setminus F}(v)$ denotes its unique neighbor in $G \setminus F$, otherwise packets get stuck at v . This case can be thought as a special case of the link-circular forwarding. If $v \in V' \cup \{s\}$ has $\Delta_{G \setminus F}(v) = 3$, i.e., $\Delta_G(v) = \Delta_{G \setminus F}(v)$ and $F_v = \emptyset$, a *non-link-circular* forwarding function at v must imply $\exists x, y \in N_{G \setminus F}(v) : \pi_v(x) = y$ and $\pi_v(y) = x$, where $N_{G \setminus F}(v) = \{x, y, z\}$ are neighbors of v in $G \setminus F$. However, a non-link-circular forwarding function cannot be 2-resilient if the only $s - t$ path remained in $G \setminus F$ has to go through the link $\{v, z\}$. For example, when $F = \{\{s, u_0\}, \{v_2, v_3\}\}$ and $\Delta_{G \setminus F}(v_1) = 3$, if a non-link-circular forwarding function has $\pi_{v_1}(v_0) = v_2$ and $\pi_{v_1}(v_2) = v_0$, then a packet starting at s cannot approach t anymore even if $s - t$ is connected via $\{v_1, v_4\}$. A similar argument can be established if $\pi_{v_1}(v_0) = v_4$ and $\pi_{v_1}(v_4) = v_0$, and $F = \{\{s, u_0\}, \{v_4, v_3\}\}$. Moreover, for each $v \in V' \cup \{s\}$ having $\Delta_{G \setminus F}(v) = 2$, a non-link-circular forwarding function at v must imply $\exists x \in N_{G \setminus F}(v) : \pi_v(x) = x$ for $N_{G \setminus F}(v) = \{x, y\}$, which can make v become a dead-end node, i.e., a packet cannot traverse from one neighbor of v to the other neighbor to approach t anymore. Therefore, each $v \in V' \cup \{s\}$ must have a link-circular forwarding function.

If $v \in V' \cup \{s\}$ has $\Delta_{G \setminus F}(v) = 2$, then its link-circular forwarding function is unique, i.e., from one neighbor to the other neighbor. If $v \in V' \cup \{s\}$ has $\Delta_{G \setminus F}(v) = 3$, where $F_v = \emptyset$, then there are two possible circular orderings for its neighbors $N_{G \setminus F}(v)$, i.e., one clockwise and the other counter-clockwise based on their geometric locations in Fig. 3. For example, at v_1 , the clockwise ordering of $N_G(v_1)$ is $\langle v_0, v_2, v_4 \rangle$ and the counter-clockwise ordering of $N_G(v_0)$ is $\langle v_0, v_4, v_2 \rangle$. Thus, for each $v \in V' \cup \{s\}$ that has $\Delta_{G \setminus F}(v) = 3$, its link-circular forwarding function can choose one of two options: clockwise and counter-clockwise, arbitrarily.

Fixing $\{s, u_0\} \in F$, let $\{v_2, v_3\} \in F$ (resp., $\{v_4, v_3\} \in F$) be a dynamic failure in $G[V' \cup \{s, t\}]$ if v_0 and v_9 both have the clockwise (resp., counter-clockwise) of link-circular forwarding functions. In this case, even if s, t are connected in $G[V' \cup \{s, t\}]$, a packet originated at s will enter a forwarding loop: $(v_0, v_1, v_2, v_1, v_4, v_3, v_2, v_1)$ (resp., $(v_0, v_1, v_4, v_1, v_2, v_3, v_4, v_1)$), where the dynamic failure $\{v_2, v_3\}$ (resp., $\{v_4, v_3\}$) is down only

for the first hitting but always up since then, but never traverses the link $\{v_3, v_5\}$ to arrive at t . When v_1 and v_3 have the different type, by fixing $\{s, u_0\} \in F$, even if there is no dynamic failure in $G[V' \cup \{s, t\}]$, a forwarding loop: $(s, v_0, v_1, v_2, v_3, v_4, v_1, v_0, s)$ occurs if v_1 and v_3 take forwarding functions of clockwise and counter-clockwise orderings respectively, otherwise another forwarding loop: $(s, v_0, v_1, v_4, v_3, v_2, v_1, v_0, s)$ exists. Moreover, a similar discussion can be applied when $\pi_s(\perp) = u_0$ for $F_s = \emptyset$.

Thus, no 2-resilient source-matched forwarding scheme against dynamic failures for (s, t) exists in Fig. 3. \blacktriangleleft

► **Theorem 4.** *There exists graphs for which any resilient source-matched routing that can tolerate $2k$ dynamic failures needs rewriting of at least k bits in packet headers for $k \in \mathbb{N}$.*

Proof of Theorem 4. We prove Theorem 4 by induction on k . Theorem 3 implies that there is a graph $G = (V, E)$ as shown in Fig. 3, where any 2-resilient source-matched routing for (v_s, v_t) needs rewriting at least one bit, proving the initial case of $k = 1$. We assume that there is a general graph $H = (U, E_U)$, where a $2k$ -resilient source-matched routing against dynamic failures for a source-destination pair (u_s, u_t) (resp., (u_t, u_s)) with $u_s, u_t \in U$ needs rewriting at least $k \geq 2$ bits.

Now, as illustrated in Fig. 5, we can construct another graph $G' = (V', E')$ by replacing each edge $\{v_i, v_j\} \in E$ in G with a graph $G_{i,j} = H^{i,j} \cup \{\{v_i, u_s^{i,j}\}, \{u_t^{i,j}, v_j\}\}$, where $H^{i,j} = (U^{i,j}, E_U^{i,j})$ is isomorphic to H , i.e., $U^{i,j} = \{u_\ell^{i,j} : u_\ell \in U\}$ and $E_U^{i,j} = \{\{u_\ell^{i,j}, u_o^{i,j}\} : \{u_\ell, u_o\} \in E_U\}$, and nodes $u_s^{i,j}, u_t^{i,j} \in U^{i,j}$. We note that we use v_s and s (resp., v_t and t) interchangeably in this proof.

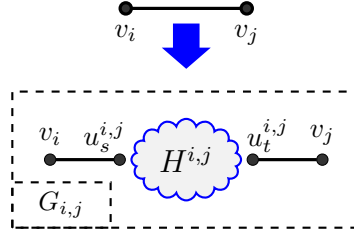
Next, we claim that any $(2k+2)$ -resilient source-matched routing for (v_s, v_t) in G' needs rewriting at least $k+1$ bits. Let $F' = F^{i,j} \cup F$ be any $2k+2$ dynamic failures in E' , where $F^{i,j} \subset E_U^{i,j}$ for $\{v_i, v_j\} \in E$ has $|F^{i,j}| = 2k$ and $F \subseteq \{\{\{v_i, u_s^{i,j}\}, \{u_t^{i,j}, v_j\}\} : \{v_i, v_j\} \in E\}$ has $|F| = 2$.

In the graph G as shown in Fig. 3, we always assume $\pi_s(\perp) = v_0$ for $F_s = \emptyset$. Let $F^* = \{e_1, e_2\}$ denote two dynamic failures in G . For example, when $e_1 = \{v_5, t\}$ and $e_2 = \{v_1, v_4\}$, the packet starting at s meets the first failure (v_5, t) , and it has to go through (v_2, v_1) back to v_1 since $\{v_1, v_4\}$ failed. Clearly, one bit in the packet header must be rewritten to inform v_1 whether the packet has already visited v_3 , s.t., v_1 can decide forwarding it to s or v_4 provided that $\{v_1, v_4\}$ is recovered. Similarly, for F^* in G' , we set $\{u_t^{5,t}, v_t\}, \{u_t^{1,4}, v_4\} \in F$ and $F^{i,j} = F^{1,2}$, we still need one bit at v_1 to indicate whether the packet has already visited v_3 . Still, to go through $H^{1,2}$ to arrive at v_1 , we need rewriting of additional k bits under failures $F^{i,j}$. Thus, we need rewriting $k+1$ bits for $2k+2$ dynamic failures in G' . For other failure cases in G , we can similarly map them to scenarios in G' . \blacktriangleleft

B Deferred Proofs of Theorems and Lemmas in Section 4

We restate the theorems and lemmas before the proofs for ease of readability.

► **Lemma 17.** *Given $\vec{\mathcal{F}}$, for each face $f_i \in \mathcal{F}$ and an ear P_j with $|E(f_i) \cap E(P_j)| \geq 2$, there cannot exist arcs $(a, b), (c, d) \in \vec{f}_i$ such that $\{a, b\}, \{c, d\} \in E(f_i) \cap E(P_j)$ with $(a, b) \in \vec{P}_j$ and $(c, d) \in \overleftarrow{P}_j$. Hence, \vec{f}_i must be consistent with either \vec{P}_j or \overleftarrow{P}_j if $|E(f_i) \cap E(P_j)| \geq 1$.*



■ **Figure 5** An illustration of constructing the graph G' in the proof of Theorem 4, where we replace each edge $\{v_i, v_j\}$ in the graph G as shown in Fig. 3 with another graph $G_{i,j} = H^{i,j} \cup \{\{v_i, u_s^{i,j}\}, \{u_t^{i,j}, v_j\}\}$.

Proof. We assume that there is a face $f_i \in \mathcal{F}$ and an ear $P_j \in \mathcal{F}$ such that $|E(f_i) \cap E(P_j)| \geq 2$, s.t., there are two distinct arcs $(a, b), (c, d) \in \vec{f}_i$ that satisfies $\{a, b\}, \{c, d\} \in E(f_i) \cap E(P_j)$, $(a, b) \in \vec{P}_j$ and $(c, d) \in \overleftarrow{P}_j$, and then we prove the theorem by a contradiction.

For a plane graph G , removing any edge preserves planarity [12, 33]. Let $G_i := \bigcup_{\ell \leq i} P_\ell$, $i \leq \kappa$, denote the plane subgraph after inserting the ear P_i . By the definition of ear-decomposition, only the endpoints of P_{i+1} lie in G_i , so adding P_{i+1} is equivalent to inserting an edge into G_i (this edge can be a self-loop on the endpoint of P_{i+1} if P_{i+1} is a closed ear). Moreover, after inserting P_{i+1} , the graph G_{i+1} is 2-edge-connected, and by Lemma 6, at least one new face is created in G_{i+1} compared to G_i .

Since both $G_i \subseteq G_{i+1}$ and G_{i+1} are plane, adding P_{i+1} (treated as an edge) can only split a face of G_i (including outer face) into two faces in G_{i+1} , otherwise, G_{i+1} would not be plane. It implies that the region of any face of G_{i+1} must be only contained in a face of G_i .

Consider $G_j := \bigcup_{\ell \leq j} P_\ell$ with an edge $P_j \in \mathcal{P}$ that satisfies $|E(f_i) \cap E(P_j)| \geq 2$. Hence, P_j must divide some face of G_{j-1} into two faces $f_i^j, f_j^j \in \mathcal{F}(G_j)$, sharing P_j as common boundary. Right-routing along \vec{f}_i^j and \vec{f}_j^j induces the directed paths \vec{P}_j and \overleftarrow{P}_j , respectively. As further ears are added, faces f_i^j and f_j^j may be further subdivided.

Now suppose there exist arcs $(a, b), (c, d) \in \vec{f}_i$ with $\{a, b\}, \{c, d\} \in E(f_i) \cap E(P_j)$, where $(a, b) \in \vec{P}_j$, $(c, d) \in \overleftarrow{P}_j$ and $f_i \in \mathcal{F}(G)$. Then the region of f_i (unbounded if $f_i = f_\infty$) would intersect both regions of f_i^j and f_j^j in G_j , contradicting the planarity of G .

Hence, \vec{f}_i must be consistent with either \overleftarrow{P}_j or \vec{P}_j . Specifically, if $E^\cap(f_j, P_j) \neq \emptyset$, then either $|E^\cap(f_j, P_j)| = |E(\vec{f}_i) \cap E(\vec{P}_j)|$ or $|E^\cap(f_j, P_j)| = |E(\vec{f}_i) \cap E(\overleftarrow{P}_j)|$, where $E^\cap(f_j, P_j) := E(f_i) \cap E(P_j)$, $E(\vec{f}_i) \cap E(\vec{P}_j)$ (resp., $E(\vec{f}_i) \cap E(\overleftarrow{P}_j)$) implies the common arcs of \vec{f}_i and \vec{P}_j (resp., \overleftarrow{P}_j). ◀